Um Projeto em Linguagem LOGO para elicitar a Produção de Estruturas Relativas em Português*

Margarete Axt**

RESUMO

Este trabalho preliminar, utilizando a linguagem LOGO de programação, constitui um "kernel" de representação do conhecimento das quatro estruturas básicas concernentes ao processo de relativização, em Língua Portuguesa. A utilização da Linguagem LOGO deu-se no intuito de explorar as suas potencialidades e seus limites enquanto linguagem voltada para a Inteligência Artificial. Nesse sentido, construímos duas pequenas gramáticas de cláusulas definidas, inspiradas nas idéias de N. Rowe (Roselló, 86), para a montagem e análise de orações, a par de um conjunto de procedimentos que interage como sujeito, fazendo-lhe perguntas que possam completar o aprendizado do programa sobre o conhecimento do mesmo. O objetivo do '"kernel" é possibilitar a construção de um sisitema mais complexo que possa acompanhar a aquisição de estruturas relativas pela criança.

INTRODUÇÃO

Este trabalho preliminar, utilizando a linguagem LOGO de programação, constitui um "kernel" de representação do conhecimento

---

* Apoio CNPq. Agradeço à Dra. Rosa Maria Viccari, professora do Curso de Pós-Graduação em Ciências da Computação (UFRGS), que colaborou na realização deste trabalho através do seu incentivo, da sua pronta orientação e significativas contribuições na área de domínio.

** Pesquisadora do Laboratório de Estudos Cognitivos do Departamento Psicologia (LEC/UFRGS) e doutoranda do CPGLL-PUCRS.

das quatro estruturas básicas concernentes ao processo de relativização, em Língua Portuguesa. Sua construção fundamenta-se nos resultados de pesquisa sobre a aquisição dessas estruturas, que vêm sendo obtidos por Axt (89; 90).

Duas variáveis são da maior importância na descrição das orações relativas do tipo SS/SO/OS/OO: 1) uma é a posição ocupada pela oração relativa, dentro da oração principal, e que chamaremos de "encaixamento". Essa posição depende da função sintática do sintagma nominal (SN) antecedente ao qual a relativa está ligada. Se o SN antecedente ocupa a posição de sujeito, o encaixamento da relativa intercala a matriz, interrompendo-a – temos aí o que se chama de "encaixamento central" *(centre embedded)*. Se o SN antecedente ocupa a posição de objeto (preposicionado ou não), temos, então, uma "expansão à direita"; 2) a outra variável diz respeito à função sintática desempenhada pelo SN anafórico dentro da oração relativa, o que é chamado de "foco". As combinações possíveis entre essa duas posições (antecedente a anafórico) dão origem aos quatro tipos básicos de que falamos e sobre os quais se concentram a maior parte das investigações sobre o processo de relativização:

(1)	Encaixamento posição SN anteced.	Foco posição SN anaf	Exemplos
(1a)	(SS)sujeito	sujeito	O gato que mordeu o cão comeu o rato
(1b)	(SO)sujeito	objeto	O gato que o cão mordeu caçou o rato
(1c)	(OS)objeto	sujeito	O gato mordeu o cão que caçou o rato
(1d)	(OO)objeto	objeto	o gato mordeu o cão que o rato assustou

Nosso interesse em torno dessas estruturas, justifica-se, em particular, pelo fato de o processo de relativização:

1) jogar um papel central não só nas línguas naturais, como também em linguagens computacionais voltadas para a Inteligência Artificial: a sua sintaxe permite a recursão, um dos fatores da criatividade ilimitada das línguas naturais, também amplamente explorada, com essa mesma finalidade, na construção de linguagens em Inteligência Artificial(1);

2) poder jogar um papel central nos estudos de aquisição da língua natural ao permitir o acompanhamento do progresso gradual da criança, através de várias etapas, no sentido da superação de dificuldades estruturais (tipos de orações relativas) e semântico-conceituais (atribuição de referente a um SN vazio e articulação da relação pressuposição-asserção em algumas relativas).

Do ponto de vista desenvolvimentista, há evidências de uma construção progressiva do processo de relativização, sujeita a estágios, os quais a criança alcança gradativamente, procedendo a sucessivas coordenações inferenciais implícitas (Axt, 90). Na falta de uma competência lingüística específica (no que diz respeito ao conhecimento estrutural do aspecto em questão independentemente da existência de variáveis contextuais ou do suporte funcional e organizacional do contexto) e enquanto o processo de aquisição desses elementos estruturais não se encontra consolidado e organizado internamente, a criança faz uso de um sistema de estratégias alternativas, impondo, ao processo de ordenação frasal, a interpretação que faz das relações semântico-categoriais e gramaticais superficiais abstraídas do contexto informacional (de)codificado (Axt, 90). Essas estratégias de segmentação da estrutura superficial (chamadas por Bowerman (79) de estratégias para segmentar estruturas oracionais complexas, e chamadas por Hamburger & Crain (82) de operações de "juntar pedaços" (patch up operations)) deixariam de ser ativadas a partir do momento em que o falante passasse a dominar a sintaxe particular, com base no conhecimento estrutural, sobrepondo-se este a quaisquer outros processos 'ad hoc'. Isso significa que, dado o tipo de contexto adequado (ou as condições de felicidade) para a atualização de uma certa estrutura sintática, essa deveria se impor na (de)codificação lingüística (Hamburger & Crain, 82). O estudo, pelo lingüista, de "erros de segmentação sintática e interpretação semântica" ou estratégias de processamento da informação lingüística têm o seu valor ao indicar o limite máximo da competência infantil em determinado momento, sem que se deva confundir, segundo os autores citados, de modo algum, aquisição (implicando competência). Ao assumir a hipótese da aquisição por partes *(piecewise acquisition)*, o lingüista, segundo Hamburger & Crain (82), deverá tentar desdobrar o quanto possível cada regra em suas sub-regras, quando da montagem do experimento, no intuito de precisar ao máximo tal limite de competência. Nesse sentido, foi possível estabelecer, de acordo com Axt (90), um conjunto característico de "vetores" (MacWhinney, 82), ou traços semântico-categoriais e sintático-funcionais em torno dos quais se organiza a aquisição do processo de relativização:

a) o vetor sintático-posicional determinante da posição ocupada pela oração relativa dentro do período oracional composto (encaixe ou expansão à direita);

b) o vetor funcional determinante das funções – relacional e/ou pronominal – desempenhadas pelo nexo "que" ao nível intra e interoracional;

c) o vetor semântico-categorial – em torno do qual se organiza o padrão de construção frasal –, 'sujeito de ação'/'sujeito temático'(2), submetido a noção de agentividade e ao qual se encontra associada a noção da dimensão temporal e a articulação da relação asserção/pressuposição.

Os nossos objetivos, na construção desse sistema, constituíram-se na tentativa de: a) viabilizar em linguagem LOGO, um sistema de aprendizagem do estágio atual do conhecimento estrutural da criança, no que diz respeito ao processo de relativização; b) organizar tal sistema de modo a privilegiar o conjunto de vetores característicos apontados por Axt (90) como fundamentais no processo de aquisição das estruturas relativas; c) criar um contexto lingüístico-estrutural (sem gráficos), razoavelmente adequado ao uso dessas estruturas (condições de felicidade), de modo a provocar a sua elicitação.

Permeou este trabalho a hipótese de que o fato de apresentar à criança, como estímulo inicial, estruturas oracionais simples (períodos absolutos), em pares, sempre com um SN em comum – o SN anafórico ou SN antecedente em aberto, apenas marcada a sua posição, podendo ser definido pelo próprio sujeito –, caracterizaria, de certo modo, a necessidade de encaixamento oracional: a marcação, apenas, da posição do SN, ou anafórico ou antecedente, antecipando a necessidade de encaixamento oracional, no sentido amplo (encaixamento central e expansão à direita), tornar-se-ia interessante, do ponto de vista da verificação do conhecimento do sujeito, na medida em que se poderia observar a sua compreensão, ou não, do indício; a sua resposta refletiria, então, o quanto essa construção se encontra, ou não, ativada enquanto mecanismo lingüístico de codificação verbal da noção de referência anafórica. Por outro lado, se a sensibilidade lingüística infantil ainda não tivesse despertado para esse tipo particular de construção, a pista *(cue)* oferecida à criança não surtiria efeito e ela sentir-se-ia confortável construindo um enunciado compatível com o grau de consciência e de competência lingüística disponível.

As páginas subseqüentes dedicam-se à descrição sucinta do sistema apresentado ao final, em anexo.

DESCRIÇÃO DO SISTEMA

A utilização da Linguagem LOGO deu-se no intuito de explorar as suas potencialidades e seus limites enquanto linguagem voltada para a Inteligência Artificial(3). Nesse sentido, construímos, adaptando de Roselló (86), duas pequenas gramáticas de cláusulas definidas ou DCGs (4), inspiradas nas idéias de N. Rowe (Roselló, 86) – uma, para a construção de duas orações simples absolutas, com um SN em comum, que pode ser relativizado, posteriormente; e uma, para a análise da estrutura construída que tanto pode ser uma estrutura relativa como uma coordenada assindética, ou sindética ou ainda uma estrutura alternativa particular. Os procedimentos (Roselló, 86) que manipulam essa duas DCGs e as suas respectivas bases de dados exploram, essencialmente, a programação funcional e modular (5) com suporte especial na noção de recursividade (6), implicando em encaixamentos sucessivos.

Dessa maneira, o bloco 1 de procedimentos (reconhecidos por terem à frente o n. 1) é o responsável pela construção de estruturas oracionais simples, em pares, sempre com um SN em comum – Vejamos cada um dos procedimentos do bloco 1:

– o procedimento OI constitui a interface interativa inicial, define algumas variáveis de valor global e armazena o nome do sujeito que servirá para individualizar e resgatar toda a sua produção escrita;

– o procedimento 1SENTENÇA define a DCG e a respectiva base de dados;

– 1S é um procedimento com duas listas de entrada, substituindo, em uma determinada lista, algumas palavras (ou conjunto de palavras) por outras. Por exemplo, em [verbo][morder], substitui "verbo" por "morder"; a substituição também pode ser feita por um elemento sorteado: a lista [[[mordeu perseguiu]]] tem dois elementos que devem ser sorteados – essa indicação é feita, colocando-se colchetes duplos;

– o procedimento principal é 1CHAMAR, inicializado pelo procedimento C, que dá o valor inicial à lista de palavras sobre as quais

serão efetuadas as substituições em 1CHAMAR, dando-lhe o nome do procedimento (a gramática DCG); realiza as substituições em obediência ao "comando" 1S, no sentido "top down";

– o procedimento 1SUBSTITUIR recorre a: LISTA dada elemento por elemento, e quando encontra um elemento igual a: SIMBOLO, o substitui por: OBJETO, continuando assim até que a: LISTA esteja vazia; como às vezes deve realizar a substituição por um elemento que deve ser sorteado, é acionado o procedimento 1EXPANDIR;

– 1EXPANDIR verifica se: LISTA é uma lista com colchetes duplos e em caso afirmativo sorteia um dos elementos, chamando o procedimento 1SORTEAR;

– 1NOMEAR controla o número de execuções, individualizando cada uma delas mediante a atribuição de variáveis;

– 1INTERAÇÃO representa a segunda interface interativa, convidando o sujeito a produzir uma estrutura complexa com base em duas estruturas simples apresentadas pelo programa.

O bloco 2 de procedimentos (reconhecido pelo n.2 à frente de cada procedimento) é o responsável pela análise e classificação da estrutura produzida pelo sujeito – o analisador sintático. Aqui as substituições deverão ser programadas no sentido inverso das do bloco 1:

– o procedimento principal é 2ANALISAR (novamente uma DCG), que toma como entrada a: LISTA de palavras escrita pelo sujeito e vai procedendo às substituições no sentido "bottom up";

– para realizar as substituições, é necessário definir um "comando" 2R que efetue as substituições, do mesmo modo que 1S, sendo nisso auxiliado por 2RTODOS;

– 2RTODOS chama o procedimento 2EXPANSÃO que deve manipular uma lista com vários elementos concorrentes. Para isso necessita do auxílio do procedimento 2SEPARAÇÃO, que cria sublistas manipuláveis dentro de uma lista;

– 2RTODOS também chama 2TROCA, responsável pela substituição da lista de elementos por um elemento só (hierarquicamente mais alto): OBJETO;

– o procedimento 2COINCIDEM? verifica se as duas listas de entrada são de forma a que a: LISTA2 coincida com os primeiros elementos da: LISTA1;

– o procedimento 2SUPRIME vai eliminando os: N primeiros elementos de: LISTA.

Se os procedimentos dos blocos 1 e 2 são principalmente voltados para a programação funcional, os do bloco 3 já se caracterizam por um tipo de programação mais procedural. O seu papel é de verificar qual a classificação da estrutura produzida e de, com base na avaliação feita, interagir como sujeito, fazendo-lhe perguntas com o intuito de completar o aprendizado do programa sobre o conhecimento desse sujeito. Nesse sentido, cada um dos procedimentos, fora o de avaliação da classificação, é pautado por uma hipótese específica sobre o que parece ser o conhecimento do referido sujeito:

– 3ESTRUTURA avalia a classificação da estrutura produzida pelo sujeito e que foi analisada pelo bloco 2;

– se a estrutura em questão for uma relativa, o programa deseja saber se o sujeito tem consciência metalingüística a respeito carga semântica do QUE anafórico e procura testar o conhecimento da noção de agentividade (3AGENTE) e da noção de pressuposição-asserção (3TEMPO), veiculadas pela construção relativa e representadas no QUE anafórico;

– se a estrutura produzida não for relativa, o programa tenta saber se tal construção pode eventualmente ser ativada, existindo no repertório lingüístico do sujeito; isso é feito, inicialmente, apenas perguntando se há outras formas se estruturar as orações simples numa composição única, complexa (3OUTROJEITO); somente após terem sido esgotadas todas as possibilidades, por parte do sujeito, o programa sugere a composição de uma estrutura relativa (3USARQUE), o que pode, ou não, ter aceitação, dependendo do estágio de desenvolvimento da mesma;

– o procedimento 3RESULTADOOUTROJEITO avalia essas produções subseqüentes, com base na análise efetuada pelo bloco 2, procurando verificar se as mesmas chegam a caracterizar-se pelo processo de relativização;

– 3REINICIAR constitui a última interface interativa, remetendo o programa novamente ao bloco 1 para outro sorteio de um par de frases simples; ao mesmo tempo individualiza toda a lista de produções do sujeito testado para posterior resgate.

A par da simplicidade do programa, apenas um "kernel", e da simplicidade das DCGs e respectivas bases de dados, inviabilizando a produção e o reconhecimento de maior número de estruturas e de frases com objetos e ações diversificadas, ficaram pendentes alguns problemas:

– o programa não detecta erros cometidos pelo sujeio durante a digitação (7);

– o programa classifica e armazena a produção do sujeito, mas não modela o seu conhecimento de forma articulada (8).

NOTAS

1 – As linguagens da Inteligência Artificial (IA) caracterizam-se por simularem o não-determinismo, logo permitem a escrita de programas não-determinísticos (exploram um espaço de possíveis soluções). Este tipo de ambiente favorece o desenvolvimento de programas recursivos.

2 – Definimos "sujeito de ação" como aquele em que o agente de cada ação é também o sujeito gramatical de cada um dos verbos do enunciado; e o "sujeito temático" como aquele em que o sujeito gramatical vem a ser o nome do personagem presente (repetido) nas duas ações (Axt, 90).

3 – Baseando-se na linguagem de IA - LISP - o Departamento de Investigação em IA do M.I.T., com uma equipe dirigida por Seymour Papert, desenvolveu a linguagem LOGO. LOGO, tendo partido de uma reflexão profunda do processo educativo, é considerada como uma linguagem para aprender. Nesse sentido rompe com preconceitos anteriores, como por exemplo, de que programação é só para matemáticos, já que é considerada uma linguagem que não possui limite inferior - corroborado pelas inúmeras aplicações para crianças com problemas de aprendizagem – nem limite superior – sua estrutura modular, aliada à possibilidade de exploração da recursividade, favorece aplicações muito complexas. Constitui, por isso mesmo, uma forma de introdução dos conceitos básicos de funcionamento dos computadores, tanto quanto da programação estruturada e de conceitos básicos de IA, dada sua estrutura de lista e grande capacidade de manipulação de símbolos, semelhante a das linguagens de IA, podendo ser desenvolvidos programas "inteligentes" de resolução de problemas, representação de conhecimentos ou sistemas especialistas. LOGO tem aplicação em geometria, matemática, ciências, estudo de língua, música, etc. (Roselló, 86).

4 – Um formalismo conhecido como gramática de cláusulas definidas – baseado no trabalho de Comerauer e Kowalski –, expressa regras livres de contexto como proposições lógicas. Muitas linguagens de programação não são adequadas para a construção de gramáticas gerativas *(generators)* e de análise *(parsers)* em língua natural. Mas em PROLOG isso é particularmente interessante, através do formalismo da DCG (THE Arity/Prolog, 86). Em LOGO, é de grande interesse o trabalho de Neil Rowe no M.I.T., cujos resultados possibilitaram a criação de uma gramática definida por leis que, de modo bastante similar, também permite a geração de gramáticas ou a realização de análise sintática de orações (Roselló, 86).

5 – Pode-se definir a programação funcional como um conjunto de funções, com uma ordem de execução que se determina dinamicamente; e a modular, como a composição de um programa complexo pela construção de procedimentos simples, funcionais ou procedurais, representando unidades conceituais dispostas sucessiva e hierarquicamente (Roselló, 86).

6 – Um procedimento é dito recursivo quando se define em função de si mesmo. A importância da recursão se deve ao fato de que é possível, mediante o seu uso, reduzir um problema complexo a versões mais simples de si próprio, dando lugar a procedimentos breves e precisos que encerrem idéias complexas e de grande potência (Roselló, 86).

7 – Existem já desenvolvidos, algoritmos para a detecção e correção automática de erros ortográficos basaeados no reconhecimento de padrões. Ou seja, a palavra escrita é comparada com a(s) palavra(s) similar(es) existente(s) no léxico do programa. Com base nesta comparação é feita a correção (Viccari 89).

8 – Para modelar o conhecimento do usuário o sistema deveria ser capaz de gerar e revisar crenças sobre o conhecimento que o usuário têm a respeito do domínio. No nosso caso, o sistema deveria ser capaz de representar crenças a respeito do conhecimento que ele acredita que a criança tenha sobre estruturas relativas. Os sistemas inteligentes procuram adaptar-se aos seus usuários através da aprendizagem simbólica automática *(machine learning)*. Da capacidade de adaptação de um sistema depende grandemente a sua flexibilidade. Numa interface de língua natural escrita é fundamental a capacidade de adaptação do sistema ao estilo do seu usuário. Assim, é importante que o sistema possa aprender as palavras utilizadas no discurso do usuário, e as construções sintáticas e semânticas empregadas durante a interação. É igualmente importante que o sistema possa perceber os objetivos, os planos, as ações, as crenças e as intenções de seus usuários. Um sistema de interface dotado destas capacidades (adaptação e aprendizagem) poderá, sem dúvida, melhor servir aos objetivos para os quais foi construído. Nesta linha podemos citar sistemas como Viccari (89). Logo, um dos desafios, no desenvolvimento de sistemas transportáveis para a interpretação da língua natural, é o fornecimento da informação relevante para que o sistema possa, com êxito, vencer a barreira entre a imagem que o usuário tem sobre o domínio do discurso e a forma como a informação esta estruturada neste domínio, para processamento computacional. É nessa linha que pretendemos continuar desenvolvendo o presente sistema, por enquanto apenas uma versão muito preliminar.

BIBLIOGRAFIA

AXT, M. *O tratamento das orações relativas por uma criança de 6 anos; um estudo de caso.* POA, PUCRS, 1989 (mimeo).

_____ . *Estratégias de processamento da informação lingüística como indicador do desenvolvimento da competência infantil quanto a estruturas recursivas verbais.* POA, LEC/UFRGS, PUCRS, 1990 (mimeo). Trabalho apresentado na 42. Reunião Anual da SBPC, no Simpósio "A construção de conhecimentos e sua representação em ambiente informático", POA, UFRGS, jul. 1990. Trabalho apresentado na Jornada sobre Aquisição e Aprendizagem da Linguagem, POA, PUCRS/CEAAL, out. 1990.

BOWERMAN, M. The acquisition of complex sentences. In: FLETCHER, P. & GARMAN, M. *Language acquisitions:* studies in first language development. 2ed. Cambridge, Cambridge University, 1979.

HAMBURGER, H. & CRAIN, S. Relative acquisition. In: KUCZAJ (ed.) *Language development*, v.1. Hillsdale, N. J. Eribaum, 1982.
MacWHINNEY, B. Basic syntactic processes. in: KUCZAJ (ed.) *Language development*, v.1. Hillsdale, N. J. Eribaum, 1982.
ROSELLÓ, Luis R. *LOGO;* de la tortuga a la inteligencia artificial, Madrid, Vector, 1986.
THE Arity/Prolog programming language. Arity Corporation, Concord, Mas., 1986.
VICCARI, R. M. *Tutor inteligente para a programação lógica;* idealização projeto e desenvolvimento. Lisboa, Laboratório Nacional de Engenharia Civil, 1989. Tese de doutoramento (mimeo).

ANEXO

Listagem dos Títulos e Procedimentos de cada Bloco

```
BLOCO UM
OI C 1SENTENÇA 1CHAMAR 1S 1 SUBSTITUIR 1EXPANDIR
1MARCAROBJETO
1SORTEAR 1NOMEAR 1INTERAÇÃO

TO OI
CLEARTEXT PR [ ] PR [ ]
PR [OI, COMO VOCE SE CHAMA?] PR [ ]
MAKE "NOMECRI RL
MAKE "NOVOJEITO [ ]
MAKE "USANDOQUE [ ]
MAKE "LISTACRI ( LIST :NOMECR| )
SETCURSOR [20 20] PR [("ENTER )] CLEARTEXT PR [ ] PR [ ]
PR SE :NOMECRI "... PR [ ]
PR [...EU VOU ESCREVER DUAS FRASES...] WAIT 50
PR [ ] PR [...PARA VOCE, LOGO A SEGUIR.] SETCURSOR [20 20] PR [(
"ENTER" )] MAKE " RC CLEARTEXT PR [ ] PR [ ]
PR [AS CORES "VERMELHO", "BRANCO" E "PRETO"] PR [ ] WAIT 50
PR [...REPRESENTAM OS NOMES...] WAIT 50 PR [ ] PR [... DE TRES
BICHINHOS.] SETCURSOR [20 20] PR [( "ENTER" )] MAKE " RC
CLEARTEXT PR [ ] PR [ ]
PR [UM DOS BICHINHOS...] WAIT 50 PR [ ] PR [...DEVERA' TER O SEU
NOME REPETIDO...] PR [ ] WAIT 50 PR [VOCE ESCOLHE QUAL REPETIR.]
WAIT 50 PR [ ] PR [ ] PR [ ]
```



```
PR [ISSO SIGNIFICA QUE...] WAIT 50 PR [ ] PR [ ] PR [...ESTAMOS
FALANDO...] PR [ ] WAIT 50 PR [...DO MESMO BICHINHO.] SETCURSOR
[20 20] PR [( "ENTER" (] MAKE " RC CLEARTEXT PR [ ]
PR [ ] PR [VAMOS EXPERIMENTAR COMBINAR...] WAIT 50 PR [ ] PR [...AS
DUAS FRASES NUMA SO'?] SETCURSOR [20 20] PR [( "ENTER" )] MAKE "
RC CLEARTEXT PR [ ] PR [ ]
PR [PARA EU ESCREVER AS DUAS FRASES...] PR [ ] WAIT 50 PR
[...APERTE A TECLA...] PR [ ] WAIT 50 ( PR [...COM A LETRA "C".]
[ ] [ ] [ ] [ ] [ ] [ ] [ ] [( "ENTER" )] )
END

TO C
CLEARTEXT
PR [ ] PR [ ] PR [...PROCESSANDO...]
PR [ ] PR [ ] PR [...AGUARDE...] PR [ ] PR [ ]
MAKE "LISTASUBSTVERBO [ ]
MAKE "N 2
MAKE "PROCEDIMENTO [1SENTENÇA]
1CHAMAR :N :PROCEDIMENTO
END

TO 1SENTENÇA
1S [1SENTENÇA] [SN SV]
1S [SV] [VERBO SN]
1S [SN] [ARTIGO SUBSTANTIVO]
1S [ARTIGO] [O]
1S [SUBSTANTIVO] [[[VERMELHO PRETO BRANCO [[?]]]]]
1S [VERBO] [[[PERSEGUIU MORDEU]]]
END

TO 1CHAMAR :N :PROCEDIMENTO
IF :N = 0 [1INTERACAO 2ANALISAR :FRASECRI 3ESTRUTURA :LISTA STOP]
MAKE "LISTA :PROCEDIMENTO
RUN :PROCEDIMENTO
1NOMEAR :N :LISTA
1CHAMAR :N - 1 :PROCEDIMENTO
END
```

```
TO 1S :SIMBOLO :OBJETO
MAKE "LISTA 1SUBSTITUIR :LISTA :SIMBOLO
END

TO 1SUBSTITUIR :LISTA :SIMBOLO
IF EMPTYP :LISTA [OP :LISTA]
TEST FIRST :SIMBOLO = FIRST :LISTA
IFTRUE [OP 1SUBSTITUIR SE ( 1EXPANDIR :OBJETO ) BF :LISTA
:SIMBOLO]
IFFALSE [OP SE FIRST :LISTA ( 1SUBSTITUIR BF :LISTA :SIMBOLO )]
END

TO 1EXPANDIR :LISTA
IF EMPTYP :LISTA [OP [ ]]
TEST LISTP FIRST FIRST :LISTA
IFFALSE [OP SE FIRST :LISTA ( 1EXPANDIR BF :LISTA )]
IFTRUE [OP SE ( 1MARCAROBJETO ( 1SORTEAR FIRST FIRST :LISTA )
:LISTA ) ( 1EXPANDIR BF :LISTA )]
END

TO 1MARCAROBJETO :OBJETOSORTEADO :LISTA
IF COUNT :LISTASUBSTVERBO < 1 [MAKE "LISTASUBSTVERBO LPUT
:OBJETOSORTEADO :LISTASUBSTVERBO OP :OBJETOSORTEADO]
IF NOT MEMBERP :OBJETOSORTEADO :LISTASUBSTVERBO [MAKE
"LISTASUBSTVERBO LPUT :OBJETOSORTEADO :LISTASUBSTVERBO OP
:OBJETOSORTEADO] [OP 1MARCAROBJETO ( 1SORTEAR FIRST FIRST :
LISTA
) :LISTA]
END

TO 1SORTEAR :L
OP ITEM ( 1 + RANDOM COUNT :L ) :L
END

TO 1NOMEAR :N :LISTA
IF :N = 2 [MAKE "FRASE1 :LISTA PR :FRASE1 PR [ ]]
IF :N = 1 [MAKE "FRASE2 :LISTA PR :FRASE2 PR [ ]]
END
```



```
TO 1INTERACAO
WAIT 20 PR [ ] PR [ ]
PR [EXPERIMENTE, AGORA, COMBINAR...]
WAIT 20 PR [ ]
PR [...ESSAS DUAS FRASES NUMA SO'!]
PR [ ] PR [ ]
MAKE "LISTACRI LPUT ( SE "SORTEIO: :FRASE1 :FRASE2 ) :LISTACRI
MAKE "FRASECRI RL
MAKE "LISTACRI LPUT ( SE "PRODUCAO: :FRASECRI ) :LISTACRI
CLEARTEXT PR [ ] PR [ ]
PR [...PROCESSANDO...]
PR [ ] PR [ ]
PR [...AGUARDE...]
END

BLOCO DOIS
2ANALISAR 2R 2RTODOS 2TROCA 2EXPANSAO 2SEPARACAO 2COINCIDEM?
2SUPRIME

TO 2ANALISAR :FRASE
MAKE "LISTA :FRASE
2R [[[ VERMELHO PRETO BRANCO VER VE V PRE PR P BRA BR B ]]] [NOME]
2R [[[O PELO POR]]] [DETERMINANTE]
2R [[[ELE MESMO]]] [PRONOME]
2R [[[PERSEGUIU MORDEU (ESTA' PERSEGUINDO) (ESTAVA PERSEGUINDO
) ( ESTA' MORDENDO ) ( ESTAVA MORDENDO ) ( FUGIU DO ) ( ESTAVA
FUGINDO DO ) ( ESTA' FUGINDO DO ) ESCAPOU ( ESTAVA ESCAPANDO ) (
ESTA' ESCAPANDO ) PERSEGUINDO MORDENDO]]] [VERBO1]
2R [[[( E' PERSEGUIDO ) ( FOI MORDIDO ) ( E' MORDIDO ) PERSEGUIDO
( FOI PERSEGUIDO ) MORDIDO]]] [VERBO2]
2R [[[, E DEPOIS [LOGO DEPOIS] MAS POREM [EM SEGUIDA] [ ]]]]
[CONJ]
2R [[[QUE [O QUAL]]]] [PROREL]
2R [[DETERMINANTE NOME]] [SN]
2R [[PRONOME]] [SN]
2R [[DETERMINANTE PRONOME]] [SN]
2R [[[ ]]] [SN]
2R [[VERBO2 VERBO2 SN]] [SVPAS]
2R [[VERBO2 VERBO2]] [SVPAS]
```

```
2R [[VERBO1 VERBO1 SN]] [SV]
2R [[VERBO1 VERBO1]] [SV]
2R [[VERBO1 SN]] [SV]
2R [[VERBO2 SN]] [SVPAS]
2R [[VERBO1]] [SV]
2R [[VERBO2]] [SV]
2R [[SN PROREL SV SV]] [ORARELSS]
2R [[SN PROREL SN SV SV]] [ORARELSO]
2R [[SN SV PROREL SN SV]] [ORARELOO]
2R [[SN SV PROREL SV]] [ORARELOS]
2R [[SN SV CONJ SV]] [COORDSS]
2R [[SN SV CONJ SN SV]] [COORDTODAS]
2R [[SN PROREL SV SVPAS]] [ORARELSSPAS1]
2R [[SN SV PROREL SVPAS]] [ORARELOSPAS1]
2R [[SN SVPAS PROREL SVPAS]] [ORARELOSPAS2]
2R [[SN PROREL SVPAS SVPAS]] [ORARELSSPAS2]
2R [[SN PROREL SV SN SV]] [ORARELTOP]
2R [[SN SV CONJ SVPAS]] [COORDSSPAS1]
2R [[SN SVPAS CONJ SVPAS]] [COORDSSPAS2]
2R [[SN SV CONJ SN SVPAS]] [COORDTODASPAS1]
2R [[SN SVPAS CONJ SN SVPAS]] [COORDTODASPAS2]
2R [[SN SVPAS CONJ SV]] [COORDSSPAS1]
2R [[SN SVPAS CONJ SN SV]] [COORDTODASPAS1]
END

TO 2R :SIMBOLOS :OBJETO
2RTODOS 2EXPANSAO :SIMBOLOS :OBJETO
END

TO 2RTODOS :SIMB :OBJETO
IF EMPTYP :SIMB [STOP]
MAKE "LISTA 2TROCA :LISTA FIRST :SIMB
2RTODOS BF :SIMB :OBJETO
END

TO 2TROCA :LISTA :SIMBOLOS
IF COUNT :LISTA < COUNT :SIMBOLOS [OP :LISTA]
TEST 2COINCIDEM? :LISTA :SIMBOLOS
IFTRUE [OP SE :OBJETO 2TROCA (2SUPRIME COUNT :SIMBOLOS :LISTA
```



```
:SIMBOLOS]
IFFALSE [OP SE FIRST :LISTA (2TROCA BF :LISTA :SIMBOLOS)]
END

TO 2EXPANSAO :SIMBOLOS
IF EMPTYP :SIMBOLOS [OP [ ]]
TEST LISTP FIRST FIRST :SIMBOLOS
IFFALSE [OP FPUT FIRST :SIMBOLOS (2EXPANSAO BF :SIMBOLOS)]
IFTRUE [OP SE (2SEPARACAO FIRST FIRST :SIMBOLOS) (2EXPANSAO BF
:SIMBOLOS)]
END

TO 2SEPARACAO :L
IF EMPTYP :L [OP [ ]]
OP FPUT (LIST FIRST :L) 2SEPARACAO BF :L
END

TO 2COINCIDEM? :LISTA1 :LISTA2
IF EMPTYP :LISTA2 [OP "TRUE]
TEST FIRST :LISTA1 = FIRST :LISTA2
IFTRUE [OP 2COINCIDEM? BF :LISTA1 BF :LISTA 2]
IFFALSE [OP "FALSE]
END

TO 2SUPRIME :N :LISTA
IF :N < 1 [OP :LISTA]
OP 2SUPRIME :N - 1 BF :LISTA
END

BLOCO TRES
3ESTRUTURA 3AGENTE 3SORTEAR 3TEMPO 3OUTROJEITO 3USARQUE
3RESULTADOOUTROJEITO 3REINICIAR

TO 3ESTRUTURA :LISTA
MAKE "LISTACRI LPUT (SE "CLASSIFICACAO :LISTA) :LISTACRI
IF (OR (:FRASECRI = NOVOJEITO) (:FRASECRI = :USANDOQUE))
[3RESULTADOOUTROJEITO :LISTA STOP]
IF (OR (:LISTA = ORARELSS]) (:LISTA + [ORARELOO]) (LISTA =
[ORARELSO]) (:LISTA = [ORARELOS]) (:LISTA = [ORARELTOP]) (:LISTA
```

```
= [ORARELSSPAS1]) (:LISTA = [ORARELSSPAS2]) (:LISTA =
[ORARELOSPAS1]) (:LISTA = [ORARELOSPAS2])) [3AGENTE 3TEMPO
3REINICIAR] [3OUTROJEITO]
END

TO 3SORTEAR :L
OP ITEM (1 + RANDOM COUNT :L) :L
END

TO 3AGENTE
PR [ ] PR [ ]
PR 3SORTEAR [[MUITO BEM!] OTIMO! CERTO! ISSO!]
PR [ ] WAIT 50
CLEARTEXT PR [ ] PR [ ]
PR [ESTA E' A SUA FRASE:] PR [ ]
PR :FRASECRI PR [ ] PR [ ] PR [ ] WAIT 20
PR [SERA' QUE A PALAVRINHA "QUE"...] WAIT 20 PR [ ]
PR [...ESTA' SUBSTITUINDO O NOME...] WAIT 20 PR [ ] PR [...DE UM
DOS BICHINHOS?] WAIT 20 PR [ ] PR [( ESCREVA SIM OU NAO )] PR [ ]
IF FIRST RL = "SIM [PR [ ] PR [QUAL BICHINHO?] MAKE "EXPLICAGENTE
RL MAKE "LISTACRI LPUT (SE "AGENTE :EXPLICAGENTE) :LISTACRI] [PR
[ ] PR [OK! ENTAO PODEMOS CONTINUAR !] SETCURSOR [20 20] PR [(
"ENTER" )] MAKE " RC]
END

TO 3TEMPO
CLEARTEXT PR [ ] PR [ ]
PR [ESTA E' A SUA FRASE:] PR [ ]
PR :FRASECR| PR [ ] PR [ ] PR [ ] WAIT 20
PR [E AGORA ME DIGA:] PR [ ] WAIT 20
PR [COM ESSA FRASE QUE VOCE ESCREVEU...] PR [ ] WAIT 20 PR
[...VOCE ACHA QUE DA' PARA SABER...] WAIT 20 PR [ ] PR [...SE UMA
DAS ACOES ( MORDER / PERSEGUIR ) ...] WAIT 20 PR [ ] PR
[...ACONTECEU ANTES QUE A OUTRA? ( ESCREVA SIM OU NAO )]
PR [ ]
IF FIRST RL = "SIM [PR [QUAL DAS DUAS ACOES...] WAIT 20 PR [ ] PR
[...VOCE ACHA QUE ACONTECEU PRIMEIRO?...] PR [ ] PR [( MORDER OU
PERSEGUIR? (] PR [ ] MAKE "EXPLICATEMPO RL MAKE "LISTACRI LPUT (SE
"TEMPO :EXPLICATEMPO) :LISTACR|] [PR [ENTAO, VAMOS ADIANTE!]
```



```
SETCURSOR [23 23] PR [( "ENTER" )] MAKE " RC]
END

TO 3OUTROJEITO
CLEARTEXT PR [ ] PR [ ]
PR [ESTA E' A SUA FRASE:] PR [ ]
PR :FRASECRI PR [ ] PR [ ] PR [ ] WAIT 20
PR [O QUE VOCE ACHA:] PR [ ] WAIT 2O
PR [SERA' QUE TEM UM OUTRO JEITO...] WAIT 2O PR [ ] PR [...DE
COMBINAR ESSAS DUAS FRASES?...] WAIT 20 PR [ ] PR [...SEM MUDAR O
SENTIDO? (ESCREVA SIM OU NAO )] PR [ ]
IF FIRST RL = "SIM [PR [ ] PR [ENTAO EXPERIMENTE ESCREVER] PR [ ]
MAKE "NOVOJEITO RL MAKE "LISTACRI LPUT (SE "OUTROJEITO:
:NOVOJEITO) :LISTACRI MAKE "FRASECRI :NOVOJEITO 2ANALISAR
:FRASECRI 3ESTRUTURA :LISTA] [3USARQUE]
END

TO 3USARQUE
CLEARTEXT PR [ ] PR [ ]
PR [A SUA FRASE E:] PR [ ]
PR :FRASECRI WAIT 20 PR [ ] PR [ ] PR [ ]
PR [SERA' QUE DARIA...] PR [ ] WAIT 2O PR [...PARA COMBINAR ESSAS
DUAS FRASES...] PR [ ] PR [...USANDO "QUE" PARA LIGA' - LAS? (
ESCREVA SIM OU NAO )] PR [ ]
IF FIRST RL = "SIM [PR [ ] PR [ENTAO EXPERIMENTE FAZER ISSO!] PR
[ ] MAKE "USANDOQUE RL MAKE "LISTCRI LPUT (SE "USARQUE:
:USANDOQUE) :LISTACRI MAKE "FRASECRI :USANDOQUE 2ANALISAR
:FRASECRI 3ESTRUTURA :LISTA] [PR ESTA' BEM!] PR [ ] WAIT 20 PR
[VAMOS ADIANTE!] SETCURSOR [20 20] PR [( "ENTER" )] MAKE " RC
3REINICIAR]
END

TO 3RESULTADOOUTROJEITO :LISTA
IF (OR (:LISTA = [ORARELSS]) (:LISTA = ORARELLOO]) (:LISTA =
[ORARELSO]) (:LISTA = [ORARELOS]) (:LISTA = [ORARELTOP])) [PR [ ]
PR [AGORA, PODEMOS CONTINUAR!] WAIT 50 3REINICIAR] [3OUTROJEITO]
END
```

```
TO 3REINICIAR
CLEARTEXT PR [ ] PR [ ]
PR [POSSO SORTEAR MAIS DUAS FRASES...] PR [ ] WAIT 20 PR [...PARA
VOCE COMBINAR? ( ESCREVA SIM OU NAO )]
IF FIRST RL = "SIM [C] [PR [ ] PR [ENTAO ADEUS!] PR [ ] PR [FOI UM
PRAZER CONVERSAR COM VOCE.] PR [ ] PR [VOLTE SEMPRE!] DEFINE
(FIRST FIRST :LISTCRI) [[ ] [OP :LISTACRI ]]]
END
```